Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method of operating a processor comprising ~~computer instruction comprises~~:

~~a branch instruction that causes~~ directing a branch in execution of an instruction stream based on any specified value being true or false and including a token that specifies the number of instructions in the instruction stream that are after the instruction to execute before performing the branch operation.

2. (Currently amended) The method ~~instruction~~ of claim 1 further comprising ~~wherein the branch instruction includes~~ a second token ~~that~~ specifies a branch guess operation.

3. (Currently amended) The method ~~instruction~~ of claim 1 wherein ~~the~~ an optional token includes defer-i which causes the processor to execute the ith instruction following the directing ~~branch instruction~~ before performing the branch operation.

4. (Currently amended) The method ~~instruction~~ of claim 1 wherein ~~the~~ an optional token can specify, one, two or three instructions following ~~branch instruction~~ to execute before performing the branch operation.

5. (Currently amended) The method ~~instruction~~ of claim 1 wherein the instruction has a format as:

br [label#], optional_token,

where br refers to a branch operation, label# is a symbolic representation of an address to branch to and option_token specifies the number of instructions to execute before performing the branch operation specified by the br branch instruction.

6. (Currently amended) The method ~~instruction~~ of claim 1 wherein one of the optional tokens are specified by a programmer or assembler program to enable variable cycle deferred branching.

7. (Currently amended) The method ~~instruction~~ of claim 1 wherein one of the optional tokens are specified to assist an assembler program to produce more efficient code.

8. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction is a branch unconditionally or branch to an instruction at a specified label based on an ALU condition code.

9. (Currently amended) The method ~~instruction~~ of claim 1 wherein the instruction is a branch to a specific label when a specified bit is set or cleared.

10. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction is a branch instruction that causes the processor to branch to the instruction at a specified label if a specified byte in a longword matches or mismatches a byte_compare_value.

11. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction is a branch instruction that causes the processor to branch to the instruction at a specified label based on whether or not a current context is a specified context in the branch instruction.

12. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction a branch instruction that causes the processor to branch if the state of a specified state name is a selected value.

13. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction a branch instruction that causes the processor to branch if a specified signal is deasserted.

14. (Currently amended) The method ~~instruction~~ of claim 1 wherein the branch instruction further includes an additional token, a guess_branch token which causes the processor to prefetch the instruction for the "branch taken" condition rather than the next sequential instruction.

Claims 15-16. (Canceled)

17. (Currently amended) A method of operating a processor comprises:

executing a branch instruction that causes a branch operation in an instruction stream based on any specified value being true or false; and

deferring performance of the branch operation of the branch instruction based on evaluating a token that specifies a ~~the~~ number of instructions to execute before performing the branch operation.

18. (Currently amended) The method of claim 17 further comprising:

evaluating a second token that specifies a branch guess operation, which causes the processor to prefetch an ~~the~~ instruction for the “branch taken” condition rather than ~~the~~ a next sequential instruction.

19. (Original) The method of claim 17 wherein the optional token is selectable by a programmer.

20. (Original) The method of claim 17 wherein the optional token is specified by a programmer or assembler program to enable variable cycle deferred branching.

21. (Original) The method of claim 17 wherein one of the optional token is specified to assist an assembler program to produce more efficient code.